TiffDLL50

ActiveX DLL

Informatik Inc.

User Manual

May 31, 2007

Print out this manual for easy reference.

Table of Contents

Installation

Click on the setup.exe or setupex.exe file and follow the prompts.

The VB version of TiffDLL50 requires the VB Runtime file msvbvm60.dll. The standard download file includes then VB runtime file. If the file is missing please contact Informatik Inc.

Important Message

As with all programs that change files, be sure that you make a full backup of your files before you use TiffDLL. Once a file has been changed, the action cannot be undone. Also, if at all possible, use different file names for the output files (i.e. do not overwrite the source files); only delete the source files once you have verified that the output files are correct. The TiffDLL program is very powerful and versatile, but if used incorrectly it can damage or destroy your files. Test your application fully before you use TiffDLL in a production environment. TiffDLL is supplied AS IS and the developer, copyright holder, supplier assume no responsibility. Please also read the Licensing Agreement section below. Do not use the program if you do not agree with the licensing terms and the disclaimers.

Trial Version

If you are using the Trial Version, please note the following important restrictions:

1. The trial version requires that the output file name be different than the source file. This requirement has been added to ensure that the user is not accidentally deleting or damaging source files names. Still, it is important that you make a full backup of your files before you use the TiffDLL program.

2. All output images are marked ‘TRIAL VERSION TIFFDLL’.

Use of TiffDLL in your Application

TiffDLL is not a Control, it is an **ActiveX DLL** (apartment-threaded). Please consult your programming language's user and reference manual for instructions on how to integrate the ActiveX DLL into your application's code. For a **command-line driven version** of TifDLL50 please contact Informatik.

For Visual Basic applications, proceed as follows:

From the Projects (or Tools) menu, choose **References** and mark the TiffDLL50 checkbox, then click on OK.

If the list of References shows MISSING:TiffDLL50, un-check the item, click on OK, choose References again from the Projects menu, and you will find the correct TiffDLL50 reference.

In your application, insert one of the sample codes shown below (VB code). The download file also includes a SAMPLE.VBP project file. With the sample_test.exe utility you can test and experiment with the parameter strings without creating a project.

There are two methods of using the ActiveX DLL:

The **Parameter Method**: All specifications are embedded into a single parameter string in accordance with a strict syntax. The function is then called with the parameter string as the argument. This is the recommended method; it is easier to implement, and the descriptions of the parameters shown below use that method.

The **Member Method, Object Method**: The individual properties of the object are set before the function is called. The Member method is covered in detail as an appendix at the end of this manual

Parameter Method:

Sample Code for VB (using Parameter Method):
(For other programming languages, please see sample code in www.Informatik.com/files.html)

```
'(General) Declaration
Private obj As TiffDLL50vic.ClsTiffDLL50

Private Sub Form_Load()
     Set obj = New TiffDLL50vic.ClsTiffDLL50
End Sub

Private Sub Command1_Click()
     Dim result As Long
     Dim Param As String
     param = "in=c:\mydir\img1.bmp;
     out=c:\dir2\TIFFDLLTEST.TIF; format=tif/14"
     result = obj.RunTiffDLL (param)
End Sub
```

```
Private Sub Form_Unload(Cancel As Integer)
      Set obj = Nothing
End Sub
```

The function returns zero (0) if successful. If you set the Test parameter to 2 (test=2), then the number of pages processed are returned. If the return value is negative, it represents an error code (see below for a list of error codes).

Because the functions have is a very large number of parameter elements the TiffDLL deviates somewhat from the general ActiveX DLL format. You only enter ONE parameter string. The single parameter string includes all the specifications (properties). For the syntax of the parameter string please see the sections below.

For **splitting multi-page TIFF files** into single-page files, and to **combine files** into a multi-page Tiff file see the Split/Combine Files section below.

For **Printing**, see Printing section below.

Parameter Syntax

(Parameter Method)

The parameter string (arguments) consists of a number of elements, such as source file, output file, graphics format, etc. Each element is separated by a semi-colon. In turn, most parameter elements are divided into sub-components, separated by the slash symbol (/).

The TiffDLL function can handle several conversions at the same time; for example you can change the size of the image, add a text annotation and a watermark with one command. Some conversions cannot be run at the same time and you need to run two functions in sequence. An error code –9003 or –3242 generally indicates that you are using a combination of conversions that is incompatible. Also, if you are using color images you may be restricted to the number of transformations within one function call.

Example of a simple parameter string:

```
In=c:\dir1\filex.tif;out=c:\dir2\TIFFDLLTEST.TIF;format=tif
/14;text=0/2/1/Sample
```

This sample parameter string specifies the following: File c:\dir1\filex.tif will be opened and saved as file c:\dir2\filez.tif, as a TIFF format 14. The text ‘Sample’ will be annotated at the top, right-adjusted, opaque. For now, don’t worry about the codes; everything is explained below. Just note how the parameter elements are separated by semi-colons and that many of the parameter elements have

sub-items separated by the slash symbol (/). Note also, that each parameter elements has a name tag followed by an equal sign (=). There must not be a space between the name tag and the equal sign. 'In=' is correct, but 'in =' is incorrect.

To summarize:

- The parameter string has many parameter elements.
- The order of the parameter elements is irrelevant.
- Most parameters are optional. Only the parameter elements that are relevant need to be included, but you can include blank parameter elements.
- Each parameter elements has a name tag followed by an equal sign (=).
- There must never be a space between the name tag and equal sign, otherwise spaces are irrelevant.
- The strings are not case sensitive. In= is the same as IN= or in=.
- Many parameter elements have sub-items, separated by the slash symbol (/). All leading sub-items must be entered, but lagging items my be omitted. Blank sub-items must also be separated by a slash symbol.
- Text strings (for example in text=) should not be enclosed in quotes.
- Semi-colons always separate the parameter strings; so never use a semi-colon inside a text string (for example in text=).

For **debugging**, add a test=1; to the parameter string, like:

```
test=1; in=c:\file.tif;out=c:\TIFFDLLTEST.TIF;format=tif/14
```

Printing is done with the '`out=`' parameter. Please see also Printing section below.

To retrieve the **properties** of images (number of pages, width, height, resolution, color, etc., use the '`out=`' parameter.

To retrieve the **software license number**, run the function in test mode, like:

```
test=1
```

Please note that when you convert a graphics file memo-type tags are not retained and are lost.

Parameters

If you use the Parameter Method, each parameter element must be preceded with the label followed by the equal sign (no space between the label and the equal sign).

If you use the Object Method, do not use the label and the equal sign; enter only the value string.

test=

Example:
```
test=2;
```

If the Test parameter is set to 2 (test=2) the conversion will be executed and the function will return the number of processed pages. If the return value is negative, the return code represents an error code.

in=

Example:
```
in=c:\somedir\somefile.tif;
```

This parameter specifies the source document. The entry must be a fully-qualified path name of an existing file. Do not use a semi-colon character in the file name.

out=

Examples:
```
out=c:\somedir\somefile.tif;
out=c:\somedir\somefile.tif;
out=c:\somedir\xfile[+001].tif;
out=print/ 6/1/1;
out=info_w;
```

(If you are using a TRIAL VERSION, the output file (out= parameter) must gave a different name than the source name (in= parameter). This is for precautionary reasons so that when testing the DLL you are not accidentally overwriting the source files.

This parameter specifies the output file, print specifications or file information specifications. Do not use a semi-colon character in the file name.

To save a file, the entry must be a fully-qualified path name and the folder (directory) must exist. You must have read/write permission for the specified folder.

To split a multi-page TIFF file into single-page files (TIFF or other graphics formats) use the numeric placeholder [+0001]. Include a sufficient number of leading zeros to accommodate all the pages.

To **print** the reformatted image, use the 'print' specifier. The print specifier consists of the keyword 'print' followed by up to six (6) optional print specification values. For instructions, please see Printing section below.

To return the detail of the **image file properties** use one of the following keywords:

- Out=info1 (or info_p) returns the number of pages in the image file
- Out=info2 (or info_w) returns width of image in pixels
- Out=info3 (or info_h) returns height of image in pixels
- Out=info4 (or info_x) returns xresolution (horizontal resolution)
- Out=info5 (or info_y) returns yresolution (vertical resolution)
- Out=info6 (or info_c) returns color depth (bits-per-pixel)
- Out=info7 (or info_t) returns Tiff compression code

    1 No compression
    2 CCITT Group 3
    3 Fax compatible Group 3
    4 Fax compatible Group 4
    5 LZW
    6 JPEG
    7 JPEG
    32773 PackBits

pages=

Examples:

```
pages=0-5;
pages=5;
pages=5-7;
pages=15-9999;
```

For multi-page graphics files you can specify a range of pages to be processed. To specify an unknown last page, use a very high number, e.g. 9999.

You can obtain the number of pages in a TIFF file with the `Out=info_p` parameter string (see section on the `Out=` parameter above.

save=

Example:
`save=1;`

This parameter specifies how the output files are saved if the file name already exists.

The parameter values are:

- 0=Overwrite without warning
- 1=Append as additional page

Generally, you should only append to TIFF files previously created by this software. If you append to TIFF files created by other software, carefully review the resulting TIFF. In any case make sure that you have made a backup of such TIFF files.

format=

Example:
`format=tif/14;`

The Format parameter string is optional. If not specified, the extension name of the output file will determine the format type and the compression type will be inherited or adapted from the source file, if appropriate.

The Format parameter consists of two sub-items: the format type and the compression type.

The available format strings are:

(For processing TIFF files with a large number of pages use codes tif/51, tif/54 or tif/53 (Tifftek32 is version 6.1 or higher and for TiffDLL50 version 5.61 or higher only)).

Format=tif/0	(uncompressed TIFF, multiple strips)
Format=tif/1	(TIFF/LZW, multiple strips)
Format=tif/2	(TIFF Packbits, multiple strips)
Format=tif/3	(TIFF 3, multiple strips)

Format=tif/4	(TIFF 4, multiple strips)
Format=tif/5	(TIFF CCITT 3, multiple strips)
Format=tif/10	(uncompressed TIFF, single strip)
Format=tif/11	(TIFF/LZW, single strip)
Format=tif/12	(TIFF Packbits, single strip)
Format=tif/13	(TIFF 3, single strip)
Format=tif/14	(TIFF 4, single strip)
Format=tif/15	(TIFF CCITT 3, single strip)
Format=tif/23	(TIFF 3, FillOrder=2)
Format=tif/24	(TIFF 4, FillOrder=2)
Format=tif/25	(TIFF CCITT 3, FillOrder=2)
Format=tif/33	(TIFF 3, single strip, FillOrder=2)
Format=tif/34	(TIFF 4, single strip FillOrder=2)
Format=tif/35	(TIFF CCITT 3, single strip, FillOrder=2)

Format=bmp/0	Windows Bitmap, uncompressed
Format=png	PNG format
Format=pcx	PCX format
Format=tga/0	Targa, uncompressed
Format=tga/1	Targa, compressed
Format=jpg/75	JPEG, 75% quality (use range between 1-99)
Format=gif	GIF format

For TIFF Class F (Dialogic) use format code 35 (or 34), resolution 204x196 and width of 1728 pixels.

colors=

Example:
```
colors=8;
```

The colors= parameter can be used to reduce the color depths of an image (bits-per-pixel). The following values are available:

- 1 = Monochrome (black and white)
- 8 = 8-bit color (256 colors) using color palette
- -8 = 8-bit grayscale (minus 8)

Only 24-bit colors are supported for JPEG files.

If you reduce color images to monochrome you may want to use the DitherMethod (dither=). See below.

dither=

Example:
```
dither=2;
```

If a color image is reduced to monochrome (black and white) the system will apply a standard dithering method. You can change or disable dithering with a setting as follows:

- 0 = Standard scatter dithering (Default)
- 1 = Ordered dither method
- 2 = Threshold method (essentially no dithering)

Dithering can have a significant effect on performance.

size=

Example:
```
size=8/11;
```

This parameter resizes the image (both image and dimensions)

The two sub-parameter values are:

Width:
Width in inches
If 'pix' or 'p' is appended to the value (e.g. 600pix), then width is taken in pixels.
If '%' is appended to the value (e.g. 50%), then width is taken as percentage of original width.

Height:
Height in inches
If 'pix' or 'p' is appended to value (e.g. 600pix), then height is taken in pixels.
If '%' is appended to value (e.g. 50%), then height is taken as percentage of original height.

Only one sub-parameter needs to be specified (width or height); the system can calculate the other sub-parameter (retaining the original width/height relationship). If you only specify the height, make sure that you precede the height value with a slash symbol (e.g. size= / 11).

One inch is equivalent to 2.5 centimeters. Be sure you use the appropriate decimal symbols (period or commas)!

Examples:

```
Size=8/11;
Size=1000pix/1200pix;
Size=8;
Size=50%;
```

canvas=

Example:
```
canvas=8/14;
```

This parameter resizes the dimensions of the image (the 'canvas'), without resizing the inside image. This option is useful to increase the bottom margin, or to change a letter size file to a legal size file.

The three sub-parameter values are:

Width:
Width in inches
If 'pix' or 'p' is appended to value (e.g. 600pix), then width is taken in pixels.
If '%' is appended to value (e.g. 50%), then height is taken as percentage of original width.

Height:
Height in inches
If 'pix' or 'p' is appended to value (e.g. 600pix), then height is taken in pixels.
If '%' is appended to value (e.g. 50%), then height is taken as percentage of original height.

One inch is equivalent to 2.5 centimeters. For JPEG files use number of pixels instead of inches. Be sure you use the appropriate decimal symbols (period or commas)!

Color of canvas:
If set the 1, the background color will be set to black. This option is rarely used.

Examples:

```
canvas=8/14;
canvas=1000pix/1200pix;
canvas=8/14;
canvas=50%;
canvas=8/14/1;
```

res=

Example:
```
res=204x196;
res=204/196;
```

This parameter specifies the horizontal and vertical resolution for TIFF files.

The values (separated by an ‘x’ or a slash symbol) are:

- Horizontal resolution
- Vertical Resolution

move=

Examples:
```
move=1/1;
move=-1;
```

This parameter shifts the image horizontally or vertically, i.e. in increases or reduces the margins, without changing the width and height of the image. This function is useful if you need to add space at the bottom to insert a text annotation (and you have space to spare at the top). A negative value moves the image left or up; a positive value moves the image right or down.

The two sub-parameter values are specified in inches. One inch is equivalent to 2.5.centimeters. Be sure you use the appropriate decimal symbols (period or commas)!

1. Horizontal shift
2. Vertical shift

For JPEG files the values must be specified in pixels (not inches or centimeters).

crop=

Examples:
```
crop=1/2/4/5;
crop=0/0.5/4/5;
```

This parameter specifies the cropping rectangle.

The four sub-parameter values are specified in inches. One inch is equivalent to 2.5.centimeters. Be sure you use the appropriate decimal symbols (period or commas)!

1. Left offset
2. Top offset
3. Right
4. Bottom

For JPEG files the values must be specified in pixels (not inches or centimeters).

rotate=

Examples:
```
rotate=90;
rotate=1;
```

This parameter specifies the rotation angle:

- 90 = 90-degree clockwise rotation
- 180 = upside-down rotation
- 270 = 90-degree counter-clockwise rotation
- -90 = Conditional 90-degree rotation *
- -270 = Conditional 270-degree rotation *

If width (pixels) is larger than height

You can also enter
1 for a 270 degree rotation,
2 for a 90 degree rotation, and
3 for a 180 degree rotation.

If you enter a non-square angle, the command will be ignored.

Generally, rotate functions should be run alone and cannot be combined with other graphics transformation.

invert=

Example:
```
invert=1;
```

This parameter inverts the image (negative copy).

flip=

Example:
```
flip=1;
```

This parameter flips the image (vertical mirroring).

mirror=

Example:
```
mirror=1;
```

This parameter creates a mirror image (horizontal flip).

text=

Examples:
```
text=-1/1/0/Sample Text;
text=-1/1/0/Page [p] of [pp];
```

This parameter specifies the placement and text of single-line annotations. For multiple text inserts (e.g. form filling) please use the ‘multitext=’ parameter (see below), or run the ‘text=’ function several times in sequence.

The four (4) sub-parameters are:

Vertical placement (in pixels):

If the value is positive, the specified position is relative to the top of the image; if the value is negative (e.g. –12), the specified position is relative to the bottom of the image. To place text at the bottom of the image, you typically would specify a minus 1 (-1).

Alignment:

0 = Left adjust (default)
1 = Centered
2 = Right adjust

Transparency:

1 = Opaque (default)
0 = Transparent

Text:

The text can include the following placeholders/merge fields:

[D] inserts the current date in the format specified in the Windows setup (short date format).
[F] inserts the file base name.
[FF] inserts the file name, including the extension.
[FFF] inserts the full path of the file name.
[P] inserts the page number.

[F], [FF] and [FFF] will print the file names in capital letters. To print lower-case characters use [f], [ff] and [fff] instead.

Example of text sub-parameter: Hello World File: [FFF] Page: [P] Date: [D]

Example of complete text parameter: text=-12,1,1,Hello World;

If you need to insert text at a specific **horizontal location** (pixel denominated) or if you need to insert **rotated text**, use the Multitext= parameter (see below).

Do not use a semi-colon character in the text string.

multitext=

Example:
```
multitext=c:\somefile\textinsert.txt;
```

This option can be used to annotate several text fields at different locations. The parameter must specify a text file that contains all the specifications. Do not use a semi-colon character in the file name.

The specification file must be an ASCII text file with the following 6 parameter elements:

1. Horizontal offset in pixels
2. Vertical offset in pixels
3. Font
4. Transparency: 0=Transparent, 1=Opaque (default)
5. Rotation (1-360) (not supported by all fonts)
6. Text

The Font specification uses similar syntax as the Font= parameter (see below). For example 12bc uses font size 12 Bold Courier. Special font names can be specified inside parenthesis (see example below).

Example of a specification file:

```
0/0/12c/0/0/Hello World
50/0/12hb///More text
100/0/12h/0/45/Confidential
100/0/12h(Century Gothic)/0/45/Confidential
```

font=

Examples:
```
font=12buh;
font=24
font=12iu/Stencil
```

This parameter specifies the font used for the text annotation (text=).

A single-string parameter specifies both the font name, font size and font attributes.

The leading numeric value represents the font size. The suffix letters specify the font type and attributes. The letters can be entered in any order. The available suffix letters are (lower-case or upper-case):

- H = Helvetica font (default)
- C = Courier font
- R = Times Roman font
- B = Bold
- U = Underline
- I = Italic

Optionally (rarely required) you can add a second parameter element and request a specific font name.

image=

Example:
`image=100/100/1/c:\somedir\image123.bmp;`

This parameter specifies the position and bitmap image overlay (e.g. watermarks).

The four sub-parameter values are:

1. Left offset in pixels
2. Top offset in pixels
3. Transparency:
   0 = Transparent (default)
   1 = Opaque
   2 = Reversed
4. Image file name (overlay image). Must be full path name of a BMP or a TIFF file. Do not use a semi-colon character in the file name.

first=

Example:
`first=1;`

If set to 1, only the first page of a multi-page TIFF file will be converted. The remaining pages will be appended unchanged (some tags may be omitted).

DeletePg=

Example:
```
deletepg=3;
```

If the specified value is a valid page number, this page will be deleted. If you need to remove several pages, you should re-build the file by iterating (looping) thru the pages, or use the pages= parameter.

Split/Combine Files

For splitting multi-page TIFF files into single-page files, a) specify a numeric placeholder in the [+nnn] format, or b) determine the number of pages in the TIFF file and loop thru the number of pages and save each page to a unique file name. See VB code snippets below.

To combine files into a multi-page Tiff file, iterate thru each file and append the file to a common destination file name. See VB code snippet below.

Code snippets:

Partial code for splitting a multi-page file into a single-page file

```
Method 1
. . . . . .
outfile = "c:\aaa\tiffdlltest\abc[+0001].tif"
param = "in=c:\xxx.tif;out=" & outfile & ";format=tif/14"
result = obj.RunTiffDLL(param)
. . . . . .

Method 2
. . . . . .
Param1 = "in=c:\xxx.tif;out=info_p"
nrpages = obj.RunTiffDLL(Param1)
For i = 1 To nrpages
    outfile = "c:\aaa\tiffdlltest\abc" & Trim(Str(i)) & ".tif"
    param2 = "in=c:\xxx.tif;pages=" & i & ";out=" & outfile &
";format=tif/14"
    MsgBox param2
    result = obj.RunTiffDLL(param2)
Next
. . . . . .
```

Partial code for combining a single-page file into a multi-page file

```
. . . . . .
Param1 = "in=c:\xyz.tif;out=c:\aaa\tiffdlltest\combined.tif;
save=1;format=tif/14"
result = obj.RunTiffDLL(Param1)
. . . . . .
```

Partial code for combining some pages (last two pages) of a multi-page file into another multi-page file

```
. . . . . .
Param1 = "in=c:\document\3pages.tif;out=info_p"
nrpages = obj.RunTiffDLL(Param1)
For i = nrpages-2 To nrpages
    param2 = "in=c:\xxx.tif;pages=" & i &
";out=c:\aaa\tiffdlltest\dest.tif;save=1;format=tif/14"
    result = obj.RunTiffDLL(param2)
Next
. . . . . .
```

Partial code for combining a multi-page file into another multi-page file

```
. . . . . .
param = "in=c:\xxx.tif ; out=c:\aaa\tiffdlltest.tif ; save=1
;format=tif/14"
result = obj.RunTiffDLL(param)
. . . . . .
```

Printing

To print a page or a range of pages use the keyword 'print' in the `out=` parameter (see out= section above). The pages to be printed can be specified with the `pages=` parameter (see `pages=` section above). If the pages are not specified, all pages of a multi-page TIFF file are printed.

The tree main parameter strings are, all optional:

1. Image width in inches (default = 8)
   Set to minus 1 (-1) for 'Size-to-Fit'. For some printers you may need to set the value to –2 or even –3.
2. Left offset in inches (default = 0)
3. Top offset in inches (default = 0)

The printed image height is calculated by the system (scaled to specified width).
The print request is always sent to the default printer.
One inch is equivalent to 2.5 centimeters.

Examples:

```
Out=print
Out=print/ 6/1/1;
Out=print/ 6;
```

Note the slash symbol after the keyword 'print'. In sample 2 printing is done with a 1 inch left and top offset and the printed image width is 6 inches. Sample 3 will print 6 inches wide without offsets (other than the non-printable areas). Sample 1 uses the 7.25 inch default width.

Unless specified different in the `pages=` parameter (see below), all pages of a multi-page TIFF file are printed.

For more advanced settings (like paper size, bin, orientation) see the paragraphs below.

Great care must be taken to ensure that the image fits into the printable area of your printer. Keep in mind that most printers have a non-printable area on each

side of the paper (left, top, right, bottom). This means that a 8.5 inch image will not fit on a 8.5 inch paper; the printable area for most printers would be between 7.5 and 8 inches. If the image is too large for the printable area, the printing function will fail. The print size is specified in the `out=` parameter (see out= section above). You can work around the non-printable area by cropping the image by ¼ inch (for example) on all sides (see crop= function above). The cropping function can be included in the print parameter string and it will not affect the actual image file. This process will need some experimenting on your part.

For advanced printing specification, the out= parameter sub-string has three (5) additional values (fifth, sixth and seventh, eighth and ninths item in command string). These are:

Letter size:
5 for Legal size
9 for A4 size
0, 1 or blank for letter size
For other sizes please see your printer manual

Paper Bin:
2 for lower bin
3 for middle bin
4 for manual insert
0, 1 or blank for upper bin
For other paper bin codes please see your printer manual

Orientation:
2 for Landscape
0, 1 or blank for Portrait

Printer Device:
(Exact spelling of printer name)

PrintJob:
Name of print job (optional)

For example:

`In=c:\dir1\file.tif;out=print/7/0/0/`**`5/4/2/Printer_XYZ/Job9;`**

will print a legal size paper (code 5) to the Manual paper bin (code 4) in Landscape (code 2) on the printer Printer_XYZ as print job Job9

Other example:

```
In=c:\dir1\file.tif;out=print/7/0/0/////Job001;
```

If the TiffDLL50 does not close after a print job, run an 'empty' DLL50 call in an application with a code such as:

```
Public obj As TiffDLL50Vic.ClsTiffDLL50
Private Sub Form_Load()
Set obj = New TiffDLL50Vic.ClsTiffDLL50
DoEvents
Set obj = Nothing
End
End Sub
```

Printing to networked printers:

If you are printing to a networked printer and the printing fails, check the following: a) Make sure that the networked printer is correctly mapped. b) Test and see if you can print the same document to a local printer. If it prints to the local printer but not to the networked printer, it is likely a network setup problem. If it does not print to the local printer, then review again the general printing trouble-shooting. c) Reduce the scale of the print. It is possible that the networked printer has a larger non-printable area and requires a reduced print scale. d) If it still does not print, then review the Spool Settings of the networked printer (In the printer control panel, choose Detail tab in Properties window and click on Spool Settings button). Check the 'Print directly to printer' box.

Error Codes

Error code –9001 means that in TRIAL VERSION the output file name (out=) must not be the same as the source file name (in=). This is for precautionary reasons so that when testing the DLL you are not accidentally overwriting the source files.

The function returns the number of pages processed. If the return value is negative, it represents an error code. If you run in Test mode (test=1) the conversion is not executed and the function returns a zero (0).

Common Error Codes:

-3242 Combination of incompatible conversions;
split the conversion into two function calls.
-3799 Invalid printer name
-9001 In Trial version, the destination file name cannot be the same as the source file name (for precautionary reasons).
-9002 Output filename must be different than source filename.
-9003 There are incompatible functions within the command line. Run some of the functions separately.
-9004 For multipage files, the destination file name cannot be the same as the source file name.

Specific Error Codes:

The first two digits represent TiffDLL errors (mainly input and syntax errors) and the last two digits represent Tifftek32 Library error codes (mainly graphics processing errors).

TiffDLL Error Codes (first two digits of the 4-digit error code):

10.. Input file not specified
11.. Output file not specified
12.. Annotation text not specified
13.. Watermark image file not specified
20.. Unable to get image data
21.. Image data for watermark file (must be .bmp format)
22.. Load watermark file (must be .bmp format)
23.. BitBlt error
24.. Memory Allocation
25.. DrawText error (annotations)
26.. Resize image

27.. Unable to open multitext spec file
29.. TextOut multitext
31.. Cropping
32.. Save file error
34.. Resize canvas
37.. Printing
38.. Convert Bits0per-Pixel, Palette

Tifftek32 Library Error Codes (last two digits of the 4-digit error code):

00	No error
01	Range error
02	Digitizer board not detected
03	Disk full, file not written
04	Filename not found
05	Variable out of range
06	Unreadable TIFF format
08	TIFF bits per sample not supported
09	Unreadable compression scheme
10	Cannot create file (Make sure the folder exists)
11	Unknown file format
12	Image is compressed DIB
14	Insufficient memory for function
16	Unreadable PCX format
17	Unreadable GIF format
18	Print error
19	Scanner error
25	Unreadable TGA format
26	Bits per pixel not supported
27	Unreadable BMP format
33	No data from device
34	Function timed out
40	Could not lock memory (invalid handle or memory discarded)
41	Print function already executing
42	Invalid image buffer address. (See above for error –3242)
43	Unreadable JPEG format
44	Image is too complex for operation
45	Paper could not be unloaded
46	ADF lid was opened
47	ADF bin is empty
48	ADF is not connected
49	ADF is connected but not ready
50	Function not available due to missing module
51	Page not found
52	Pointer does not point to readable or writable memory

53	LZW compression/decompression not enabled (Contact Informatik)
54	Could not create TWAIN parent window
55	Could not open TWAIN Source Manager
56	Could not open TWAIN Data Source
57	TWAIN image acquisition error
58	None of the elements in two lists were equal
59	Data type mismatch
60	User canceled scan
61	TWAIN function is busy
62	File contains invalid data
63	Unreadable PNG format
64	PNG compressor error
65	No ACK from device
66	TWAIN ADF is empty
67	Stop scanning images
68	Handle not valid
69	TIFF file is in Motorola byte order
99	Invalid printer name

License Fees

Please see www.Informatik.com/tiffdll.html or contact Informatik Inc. For contact information please visit www.Informatik.com/company.html

Technical Support

Please see www.Informatik.com/support.html

Distribution to End-Users

The following files must be included with your distribution:

- TiffDLL50.DLL (must be registered in the Windows Registry)
- Tifftek32.dll
- VB6 runtime

Copyright

License Agreement

Please read this License Agreement carefully before you install and use the software. By installing and using the software you agree with the terms and conditions of this License Agreement. If you do not agree with the terms and conditions you must uninstall the software.

This License Agreement ("Agreement") is a legal agreement between Informatik Inc., Devon Pennsylvania, USA ("Licensor") and you, the user ("Licensee"), and becomes effective on the day the Licensee installs the software. This Agreement covers all materials associated with this software, including, without limitation, the downloadable files relating to this software, printed and online documentation, and any additional supporting files and programs (herein, the "Software").

1. GRANT OF LICENSE
Licensor hereby grants to you, and you accept, a nonexclusive license to use the Software on a specified number of computers/workstations owned, leased, or otherwise controlled by you for personal or business purposes, and only as authorized in this License Agreement. The Software may not be used on other computers, nor may it be used by, or transferred to, other computers over a network. Unless the Licensee purchased a server license, the Software may not be installed on a server, or on a PC/workstation that acts as a server. Hosting a batch service on a server, PC or workstation requires a server version. A site license permits the installation of the Software on an unlimited number of PC/workstations at one site for the enterprise.

The Software is used in the Licensee's own application ("Application'). The standard version of the Software allows the Licensee to distribute the Application to and install it on a maximum of twenty-five end-user PCs and/or workstations (not servers). The royalty-free version of the Software allows the Licensee to distribute the Application and install it on an unlimited number of PCs, workstations and/or servers. The Software must represent a relatively minor part of that application. Under no circumstances must the Software be used to create another toolkit, control, COM object or library, unless it is strictly used internally by the Licensee under the terms of this license, and is not sold to or used by third parties. A separate license for the Software is required for each developer.

2. LICENSOR'S RIGHTS
Licensee acknowledges and agrees that the Software is proprietary to Licensor and protected under international copyright law. Licensee further acknowledges

and agrees that all right, title, and interests in and to the Software, including associated intellectual property rights, are and shall remain with Licensor. The License Agreement does not convey to Licensee an interest in or to the Software, but only a limited right of use that may be revoked in accordance with the terms of this License Agreement.

3. OTHER RESTRICTIONS
Licensee agrees to make no more than one (1) back-up copy of the Software. Licensee agrees not to assign, sublicense, transfer, pledge, lease, rent, or share the rights assigned under this License Agreement. Licensee agrees not to reverse assemble, reverse compile, or otherwise translate the Software.

4. TERM
This License Agreement is effective when Licensee installs the Software and shall terminate only if the terms of this License Agreement are broken. Licensee agrees to destroy the Software upon termination of this License Agreement.

5. NO WARRANTY; LIMITATION OF LIABILITY LICENSEE ACKNOWLEDGES THAT THE SOFTWARE IS PROVIDED ON AN "AS IS" BASIS WITHOUT WARRANTY OF ANY KIND. LICENSOR, DISTRIBUTORS OR AGENTS MAKE NO REPRESENTATIONS OR WARRANTIES REGARDING THE USE OR PERFORMANCE OF THE SOFTWARE. LICENSOR DISTRIBUTORS and AGENTS EXPRESSLY DISCLAIM THE WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. LICENSOR, DISTRIBUTOR OR AGENT SHALL HAVE NO LIABILITY TO LICENSEE OR ANY THIRD PARTY FOR ANY LOSS OR DAMAGE CAUSED, DIRECTLY OR INDIRECTLY, BY THE SOFTWARE, INCLUDING, BUT NOT LIMITED TO, ANY INTERRUPTION OF SERVICES, LOSS OF BUSINESS, LOSS OF DATA OR SPECIAL, CONSEQUENTIAL OR INCIDENTAL AMAGES.

6. GOVERNING LAW

The legal jurisdiction of this License Agreement shall be Pennsylvania, USA.

Appendix:

Member Method, Object Method:

Instead of defining a parameter string you can use the properties of the declared object. In your application's code, if you declared TiffDLL50.ClsTiffDLL as an object (named 'obj' in our example), simply type the name of the object (obj) followed by a period (.). The programming interface will then display a list of all available properties. Pick the relevant property and define it, as in the following example:

```
obj.File_In = ″c:\somedir\somefile.tif″
obj.File_Out = “c:\tiffdlltest\filexyz.tif″
obj.Format_Out = “tif/14″
etc.
```

The syntax of the object's property string is identical to the string that you would use in the parameter (except for the label; don't use the label with the equal sign). See Parameters section below.

Using the Object method, you run the function with a blank parameter, like

```
result = obj.RunTiffDLL (“”)
```

Sample Code for VB (using Object Method):
(For other programming languages, please see sample code in www.Informatik.com/files.html)

```
‘(General) Declaration
Private obj As TiffDLL50vic.ClsTiffDLL50

Private Sub Form_Load()
     Set obj = New TiffDLL50vic.ClsTiffDLL50
End Sub

Private Sub Command1_Click()
     Dim result As Long
     obj.File_In = ″c:\somedir\somefile.tif″
     obj.File_Out = “c:\tiffdlltest\filexyz.tif″
     obj.Format_Out = “tif/14″
     result = obj.RunTiffDLL (“”)
End Sub

Private Sub Form_Unload(Cancel As Integer)
     Set obj = Nothing
End Sub
```

The properties of the object are (all string data types):

File_In (equivalent to **in=** in Parameter Method)
File_Out (equivalent to **out=** in Parameter Method)
Format_Out (equivalent to **format=** in Parameter Method)

Format_Colors(equivalent to **colors=** in Parameter Method)
DitherMethod(equivalent to **dither=** in Parameter Method)
SaveMethod (equivalent to **save=** in Parameter Method)
Size_Image (equivalent to **size=** in Parameter Method)
Size_Canvas (equivalent to **canvas=** in Parameter Method)
XYresolution (equivalent to **res=** in Parameter Method)
Image_Move (equivalent to **move=** in Parameter Method)
Image_Crop (equivalent to **crop=** in Parameter Method)
Image_Rotate (equivalent to **rotate=** in Parameter Method)
Image_Flip (equivalent to **flip=** in Parameter Method)
Image_Mirror (equivalent to **mirror=** in Parameter Method)
Text_Annotation (equivalent to **text=** in Parameter Method)
Text_Font (equivalent to **font=** in Parameter Method)
Text_MultiText (equivalent to **multitext=** in Parameter Method)
WatermarkImage (equivalent to **image=** in Parameter Method)
Page_FirstOnly (equivalent to **first=** in Parameter Method)
Page_Range (equivalent to **pages=** in Parameter Method)
TestMode (equivalent to **test=** in Parameter Method)

Index